**Matrix Matrix Multiplication**

#include<iostream>

#include<omp.h>

#include<math.h>

using namespace std;

int main()

{

cout<<"Number of rows in first matrix: ";

int row1;

cin>>row1;

cout<<"Number of columns in first matrix: ";

int col1;

cin>>col1;

int row2 = col1;

cout<<"Number of columns in second matrix: ";

int col2;

cin>>col2;

cout<<endl;

cout<<"Matrix 1:"<<endl;

int mat1[row1][col1];

for(int i=0; i<row1; i++)

{

for(int j=0; j<col1; j++)

{

mat1[i][j] = rand()%10;

cout<<mat1[i][j]<<" ";

}

cout<<endl;

}

cout<<endl;

cout<<"Matrix 2:"<<endl;

int mat2[row2][col2];

for(int i=0; i<row2; i++)

{

for(int j=0; j<col2; j++)

{

mat2[i][j] = rand()%10;

cout<<mat2[i][j]<<" ";

}

cout<<endl;

}

cout<<endl;

cout<<"Multiplication: "<<endl;

int mat3[row1][col2];

#pragma omp parallel for

for(int i=0;i<row1;i++)

{

#pragma omp parallel for

for(int j=0;j<col2;j++)

{

mat3[i][j]=0;

int arr[col1];

#pragma omp parallel for

for(int k=0;k<col1;k++) arr[k] = mat1[i][k]\*mat2[k][j];

int sum = 0;

#pragma omp parallel for reduction(+: sum)

for (int l = 0; l < col1; l++) sum += arr[l];

mat3[i][j] = sum;

}

}

for(int i=0; i<row1; i++)

{

for(int j=0; j<col2; j++) cout<<mat3[i][j]<<" ";

cout<<endl;

}

return 0;

}

/\*

Number of rows in first matrix: 5

Number of columns in first matrix: 5

Number of columns in second matrix: 6

Matrix 1:

1 7 4 0 9

4 8 8 2 4

5 5 1 7 1

1 5 2 7 6

1 4 2 3 2

Matrix 2:

2 1 6 8 5 7

6 1 8 9 2 7

9 5 4 3 1 2

3 3 4 1 1 3

8 7 4 2 7 7

Multiplication:

152 91 114 101 86 127

166 86 144 138 74 134

78 43 106 97 50 100

119 79 106 78 66 109

69 38 66 57 32 62

Process returned 0 (0x0) execution time : 4.553 s

Press any key to continue.

If Matrix A has dimensions 5 x 5 (5 rows and 5 columns),

and Matrix B has dimensions 5 x 6 (5 rows and 6 columns),

the resultant matrix after multiplication will have dimensions 5 x 6 (5 rows and 6 columns).

\*/